Acnet Alarm Block Support

Floating point changes

Fri, Jul 14, 2000

The internal format of Acnet alarm block information is different from its form as used by Acnet. The IRM system maintains an alarm flags word and trip count word in addition to a nominal and tolerance value. This internal format must be mapped into the Acnet alarm block standard format in the RETDAT and SETDAT handlers. This note explores the detail of how this can be done so as to support the new floating point forms of raw data.

The scheme used for the integer cases is to make sure that the data collected by the normal read routine includes the 6 bytes that includes the nominal word, the tolerance word, and the alarm flags word. A post-processing routine is called to convert this information into that specified by the Acnet alarm block standard. This is fairly straightforward. But the floating point values for nominal and tolerance are not found in the same way, or even in the same ADATA table A new FDATA table, parallel to ADATA, houses the nominal and tolerance floating point values. It is not enough to have the integer values; one must gain access to the FDATA fields, too.

The internal pointer values used for the cases has the format of a pointer to the nominal field in the ADATA table entry. To get the pointer to the parallel entry in the FDATA table, assuming that both tables use 16-byte entry sizes, one can merely add the appropriate offset to the ADATA entry field pointer to get the proper FDATA entry field pointer. This assumes having access to the internal pointer by the post-processing routine. At present, the internal pointer is not passed to the post-processing routine, but rather a pointer into a buffer that holds the nominal, tolerance and alarm flags words.

Another approach would be to expand the format of the internal pointer to two longwords, such as is done to support time-stamped data accesses and averaging. The first longword could hold the pointer to the ADATA entry nominal field, and the second could be the pointer to the FDATA entry nominal field. If the post-processing routine had access to the internal pointers structure, it could easily build the required analog alarm block. To simplify the required support, use the normal 4-byte internal pointer format.

Part of the considerations in implementing this logic in assembly code is being careful about register utilization. The RETDAT work of mapping the internal analog alarm information from the ADATA entry into the Acnet alarm block structure is done by post-processing logic that is invoked after the local alarm information has been collected. This logic modifies the data into the Acnet format. In order to gain extra registers to do this manipulation inside the PPANABL routine, preserve A1-A2 around the calls to the post-processing routine. (There will be more than one call if an array of alarm blocks for consecutive channels is requested.) Inside PPANABL, we need to use D2-D3, so preserve them inside so they are not changed upon return from PPANABL. In order to provide a pointer to the internal pointer longword for each call to PPANABL, use A1 for this purpose, advancing it by 4 bytes for each call. It already points to the array of internal pointers after the read routine has been called. Note that we can assume a 4-byte internal pointer format since no special time-stamp or averaging logic is used for the analog alarm block case.

Inside PPANABL, check for the case of the FLT bit set in the alarm flags word. If it is, we have a floating point case to deal with. Quickly check the system table directory to determine whether an FDATA table exists—support of floating point data requires it—and that its defined entry size matches that of the ADATA table. (Each table should have 16-byte entries, so that parallel entries have the same offsets within the table.) If this is so, compute the difference between the

base addresses of the two tables. Taking the internal pointer, which is actually a pointer to the nominal field of the ADATA table entry, use the computed difference to access the corresponding FDATA entry to obtain the floating point values for nominal and tolerances, and replace the D4-D5 registers with these two values. (They were preset with the integer values before this new logic.) Only keep the FLT bit set if there were no surprises in doing all this.

Inside ADJABLK, called by PPANABL, check for the FLT bit set in the register containing the alarm flags word, and if it is set, store the 32-bit floating point values rather than the integer words padded with a zero word.

Besides delivering the proper nominal and tolerance values, two additional changes must be made. For the floating point case, the nominal and tolerance value sizes must be announced in the Acnet alarm status word in the Q0-Q1 two-bit field. Instead of a value of 1, which means 2-byte data, use a value of 2, which means 4-byte data.

The final change has to do with the new support for min/max values in the alarm scanning logic. The Acnet alarm status word uses the K0-K1-K2 three-bit field for this. It must indicate 2 for min/max, rather than 0 for the nominal/tolerance.

Analogous additions must be made to support settings of analog alarm blocks via the SETDAT protocol.